(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－180025

(43)公開日　平成８年(1996)７月12日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/16	３４０　Ａ			
9/46	３６０　Ｃ			

審査請求　未請求　請求項の数５　ＯＬ　（全 13 頁）

(21)出願番号　特願平6－318105

(22)出願日　平成６年(1994)12月21日

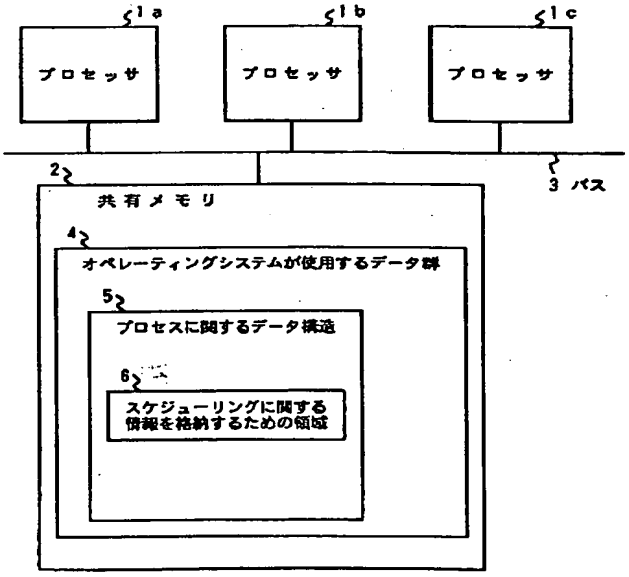
(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地

(72)発明者　岐津　俊樹
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内

(74)代理人　弁理士　鈴江　武彦

(54)【発明の名称】　スケジューリング装置

(57)【要約】
【目的】アドレス空間を共有するプロセスが同一のプロセッサで実行されるようにスケジューリングするスケジューリング装置を提供する。
【構成】複数のプロセッサ１ａ～１ｃがメモリ２を共有するマルチプロセッサシステムであって、アドレス空間を共有する複数のスレッドから構成されるプロセスを実行するマルチプロセッサシステムのスケジューリング装置において、上記スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位にデータ構造５の領域６に格納する手段と、上記プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、この領域６に格納された情報からそのスレッドが当該プロセッサにて実行可能であるか否かを判断する手段とを具備し、同一のプロセスに属するスレッドすべてを同一のプロセッサで実行させることを特徴とする。



1

【特許請求の範囲】

【請求項1】 複数のプロセッサがメモリを共有するマルチプロセッサシステムであって、アドレス空間を共有する複数のスレッドから構成されるプロセスを実行するマルチプロセッサシステムのスケジューリング装置において、

上記スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位に管理する手段と、上記プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、上記格納された情報からそのスレッドが当該プロセッサにて実行可能であるか否かを判断する手段とを具備し、同一のプロセスに属するスレッドすべてを同一のプロセッサで実行させることを特徴とするマルチプロセッサシステムのスケジューリング装置。

【請求項2】 上記管理手段は、複数のプロセス間でアドレス空間の共有が発生した際に、このプロセス群単位に上記スレッドそれぞれが実行されるべきプロセッサを示す情報を管理する手段を具備し、同一のプロセス群に属するスレッドすべてを同一のプロセッサで実行させることを特徴とする請求項1記載のマルチプロセッサシステムのスケジューリング装置。

【請求項3】 上記管理手段は、複数のプロセス群間でアドレス空間の共有が発生した際に、アドレス空間の共有を要求した側のプロセスが形成するすべてのプロセス群の情報をアドレス空間の共有を要求された側のプロセスが形成するプロセス群の情報に更新する手段を具備し、アドレス空間を共有するプロセス群に属するスレッドすべてを同一のプロセッサで実行させることを特徴とする請求項2記載のマルチプロセッサシステムのスケジューリング装置。

【請求項4】 上記管理手段は、いずれかのプロセスがアドレス空間の共有を解消した際に、このプロセスが形成するプロセス群同士のアドレス空間の共有が解消されるか否かを判定し、プロセス群同士のアドレス空間の共有が解消される場合に、これらのプロセス群各々に個別の情報を割り振る手段を具備してなることを特徴とする請求項3記載のマルチプロセッサシステムのスケジューリング装置。

【請求項5】 上記管理される情報は、上記複数のプロセッサのいずれのプロセッサで実行してもかまわない旨を示す情報を含むことを特徴とする請求項1、2、3又は4記載のマルチプロセッサシステムのスケジューリング装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、複数のプロセッサがメモリを共有するマルチプロセッサシステムに適用して好適なスケジューリング装置に係り、特にキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることによりシステム性能の向上を図ることを可能とするスケジューリング装置に関する。

2

【0002】

【従来の技術】従来の複数のプロセッサがメモリを共有するマルチプロセッサシステムにおいては、共有メモリを使用することを前提にオペレーティングシステムが作成されており、通常は、処理性能を向上させるために、メモリより高速にアクセス可能なキャッシュメモリを設け、メモリにアクセスした結果をこのキャッシュメモリに蓄積するようにしている。

【0003】しかし、プロセススケジューラは、複数のプロセス間でアドレス空間を共有している場合でも、スケジュールの際にそれを考慮することがなく、そのためにアドレス空間を共有しているようなプロセス同士が別々のプロセッサで実行されることがあり、このときに共有しているアドレス空間に書き込み等の更新が行われると、キャッシュメモリの一貫性を保持するための排他制御を行う必要が発生し、これがオーバーヘッドとなりシステム全体の効率を低下させていた。

【0004】

【発明が解決しようとする課題】上述したように、従来の複数のプロセッサがメモリを共有するマルチプロセッサシステムのスケジューリング装置においては、プロセスをスケジューリングする際に、プロセス間のアドレス空間の共有を考慮に入れることがなかったため、キャッシュメモリ及び共有メモリの一貫性を保持するためのオーバヘッドを発生させるといった問題があった。

【0005】本発明は上記実情に鑑みなされたものであり、プロセス間でアドレス空間を共有している場合に、それらのプロセスを同一のプロセッサにスケジューリングして実行することにより、キャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させ、システム性能の向上を図ることを可能とするスケジューリング装置を提供することを目的とする。

【0006】

【課題を解決するための手段】本発明は、複数のプロセッサがメモリを共有するマルチプロセッサシステムであって、アドレス空間を共有する複数のスレッドから構成されるプロセスを実行するマルチプロセッサシステムのスケジューリング装置において、上記スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位に管理する手段と、上記プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、上記格納された情報からそのスレッドが当該プロセッサにて実行可能であるか否かを判断する手段とを具備し、同一のプロセスに属するスレッドすべてを同一のプロセッサで実行させることを特徴とする。

【0007】また、本発明は、上記管理手段が、複数のプロセス間でアドレス空間の共有が発生した際に、このプロセス群単位に上記スレッドそれぞれが実行されるべきプロセッサを示す情報を管理する手段を具備し、同一

のプロセス群に属するスレッドすべてを同一のプロセッサで実行させることをすることを特徴とする。

【０００８】また、本発明は、上記管理手段が、複数のプロセス群間でアドレス空間の共有が発生した際に、アドレス空間の共有を要求した側のプロセスが形成するすべてのプロセス群の情報をアドレス空間の共有を要求された側のプロセスが形成するプロセス群の情報に更新する手段を具備し、アドレス空間を共有するプロセス群に属するスレッドすべてを同一のプロセッサで実行させることを特徴とする。

【０００９】また、本発明は、上記管理手段が、いずれかのプロセスがアドレス空間の共有を解消した際に、このプロセスが形成するプロセス群同士のアドレス空間の共有が解消されるか否かを判定し、プロセス群同士のアドレス空間の共有が解消される場合に、これらのプロセス群各々に個別の情報を割り振る手段を具備してなることを特徴とする。また、本発明は、上記管理される情報が、上記複数のプロセッサのいずれのプロセッサで実行してもかまわない旨を示す情報を含むことを特徴とする。

【００１０】

【作用】本発明の構成によれば、管理手段が、スレッドそれぞれが実行されるべきプロセッサを示す情報をプロセス単位に管理しており、プロセッサが実行待ち行列に蓄積されたスレッドを実行する際に、この管理された情報を参照して自身が実行可能なスレッドであるか否かを判定する。この判定により実行可能な具体例としては、

（１）その情報がいずれのプロセッサで実行してもかまわない旨を示している場合。

（２）その情報が自身のプロセッサを示している場合。

が挙げられる。

【００１１】この情報はプロセス単位に管理されるため、同一のプロセスに属するスレッドはすべて同一のプロセッサで実行されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【００１２】また、本発明の構成によれば、複数のプロセス間でアドレス空間の共有が発生した際に、上述した管理手段が、アドレス空間を共有するこれら複数のプロセスにより形成されるプロセス群単位にスレッドそれぞれが実行されるべきプロセッサを示す情報を管理する。

【００１３】例えば、Ａというプロセスが使用するアドレス空間について、Ｂ及びＣというプロセスが共有を要求した場合、このＡ、Ｂ及びＣのプロセスを一つのプロセス群として、このプロセス群単位に情報を管理する。

【００１４】これにより、アドレス空間を共有するＡ、Ｂ及びＣのプロセスで形成されるプロセス群に属するスレッドはすべて同一のプロセッサで実行されることになり、これらのプロセスがアドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【００１５】また、本発明の構成によれば、複数のプロセス群間でアドレス空間の共有が発生した際に、上述した管理手段が、アドレス空間の共有を要求した側のプロセスが形成するすべてのプロセス群の情報を、アドレス空間の共有を要求された側のプロセスが形成するプロセス群の情報に更新する。

【００１６】例えば、いま、Ａ、Ｂ及びＣにより構成される第１のプロセス群と、Ｘ、Ｙ及びＺにより構成される第２のプロセス群が存在し、この第１及び第２のプロセス群は、それぞれ個別のプロセッサで実行されるものとする。そして、第２のプロセス群を形成するＸのプロセスが、第１のプロセス群を形成するＡのプロセスの使用するアドレス空間の共有を要求した場合を考える。

【００１７】このとき、管理手段は、Ａのプロセスが形成する第１のプロセス群の情報と、Ｘのプロセスが形成する第２のプロセス群の情報とが等しいか否かを判断し、異なっていれば、第２のプロセス群の情報を第１のプロセス群の情報に更新する。

【００１８】この結果、Ａ、Ｂ、Ｃ、Ｘ、Ｙ及びＺのプロセスはすべて同一のプロセッサで実行されることになり、Ａ及びＸのプロセスがアドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【００１９】なお、Ｘのプロセスが、Ｙ及びＺのプロセスと形成する第２のプロセス群以外に他のプロセスとプロセス群を形成していた場合には、そのすべてのプロセス群の情報を、第１のプロセス群の情報に更新することにより、これらのオーバーヘッドを減少させる。

【００２０】また、本発明の構成によれば、いずれかのプロセスがアドレス空間の共有を解消した際に、このプロセスが形成するプロセス群同士のアドレス空間の共有が解消されるか否かを判定し、プロセス群同士のアドレス空間の共有が解消される場合に、これらのプロセス群各々に個別の情報を割り振る。

【００２１】ここでは、上述したようなＡ、Ｂ及びＣにより形成される第１のプロセス群と、Ｘ、Ｙ及びＺにより形成される第２のプロセス群がアドレス空間を共有している場合、具体的には、Ａ及びＸのプロセスがアドレス空間を共有している場合を前提として、このＡ及びＸのプロセスによるアドレス空間の共有が解消された場合を考える。

【００２２】このとき、管理手段は、この第１及び第２のプロセス群同士のアドレス空間の共有が解消されるか否かを判定する。この判定でプロセス群同士のアドレス空間の共有の解消が拒否される例としては、さらに、Ｂ及びＹのプロセスがアドレス空間を共有している場合等

5

が挙げられる。

【００２３】そして、プロセス群同士のアドレス空間の共有が解消される場合には、もはや同じプロセッサで実行すべき理由がないため、第２のプロセス群の情報を他の情報に割り振ることが可能となる。この割り振りは、プロセッサの稼働率に基づく等、所定の規則に従って行えばよい。これにより、システム全体の効率をより向上させることが可能となる。

【００２４】

【実施例】以下図面を参照して本発明の実施例を説明する。まず、図１を参照して本発明の第１実施例を説明する。図１は第１実施例に係るスケジューリング装置の概略構成を示す図である。

【００２５】図１に示すように、同実施例に係るスケジューリング装置は、プロセッサ１ａ～１ｃと共有メモリ２とがバス３を介して接続されている。また、共有メモリ２には、オペレーティングシステムが実行する際に必要となるデータ群４が格納されている。そして、このデータ群４には、プロセスに関するデータ構造５が含まれ、さらに、このプロセスに関するデータ構造５には、スケジューリングに関する情報を格納するための領域６が含まれている。このスケジューリングに関する情報を格納するための領域６には、各プロセスに属するスレッドが実行されるべきプロセッサのＩＤが格納される。また、プロセッサ１ａ～１ｃのいずれのプロセッサで実行されてもかまわない場合には、その旨を示す値が格納される。

【００２６】ここで、あるプロセッサが、実行キューに投入されたスレッドを処理しようとしたとき、そのプロセッサは、この領域６に格納された情報を参照して、自身が実行可能であるか否かを判定する。この場合、領域６に格納された情報が、

（１）いずれのプロセッサで実行してもかまわない旨を示している。

（２）その情報が自身のプロセッサを示している。

ときにそのスレッドを実行する。

【００２７】これにより、スレッドを実行すべきプロセッサの情報がプロセス単位に管理されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【００２８】次に、図２乃至図４を参照して本発明の第２実施例を説明する。図２に示すように、同実施例に係るスケジューリング装置は、プロセッサ１ａ～１ｃと共有メモリ２とがバス３を介して接続されている。また、共有メモリ２には、オペレーティングシステムが実行する際に必要となるデータ群４が格納されている。そして、このデータ群４には、プロセスに関するデータ構造５及び複数のプロセス間におけるアドレス空間の共有に関するデータ構造８が含まれ、さらに、このプロセスに関するデータ構造５には、スケジューリングに関する情報を格納するための領域６とアドレス空間の共有に関するデータ構造８を指し示すポインタ７とが含まれている。

6

【００２９】図３及び図４は、このアドレス空間の共有に関するデータ構造８を具体的に示した概念図である。ここでは、図３に示すように、システム中にプロセス９ａ、９ｂ及び９ｃがあり、それらがあるアドレス空間を共有する場合を考える。図４に示すように、これらのプロセスに対応するデータ構造は、それぞれ１３ａ～１３ｃである。そして、このプロセス群に対しprocgroup なるデータ構造１１を割り当てる。また、このプロセス群に属する各プロセスに対応してpgseg なるデータ構造１２ａ～１２ｃを割り当てる。そして、これらはリンクされた形で保持される。

【００３０】このprocgroup １１は、pgseg データ構造を指し示すためのポインタであるpg-pgseglink、これを更新する際に排他制御を行うためのロックであるpg-pgseglock、及びこのprocgroup １１に属するプロセスが実行されるべきプロセッサのＩＤが格納されるpg-cpuなるメンバにより構成される。一方、pgseg １２ａ～１２ｃには、次のpgseg 構造を指し示すためのポインタであるpgseg-link、procgroup １１を指し示すためのポインタであるpgseg-pg、及びこのpgseg が属するプロセスのデータ構造１３ａ～１３ｃを指し示すためのポインタであるpgseg-procpなるメンバで構成される。また、各プロセスのデータ構造１３ａ～１３ｃには、対応するpgsegデータ１２ａ～１２ｃを指し示すためのポインタであるp-pgsegpなるメンバが存在する。このp-pgsegpは、アドレス空間を共有していない場合には初期値が格納されている。

【００３１】ここで、図５を参照してプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明する。図５はプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャートである。

【００３２】まず、オペレーティングシステムは、procgroup データ構造１１及びpgseg データ構造１２ａ～１２ｃを割り当て（図５のステップＡ１～Ａ２）、それぞれのデータ構造を適切にリンクすることにより、各メンバの初期化を行う（図５のステップＡ３）。最後に、オペレーティングシステムは、これらのプロセス群をどのプロセッサで実行すべきかを決定し、procgroup データ構造１１のメンバであるpg-cpuにそのプロセッサＩＤを格納する（図５のステップＡ４）。

【００３３】次に、図６を参照してオペレーティングシステムがスレッドを実行する際の動作手順を説明する。図６はオペレーティングシステムがスレッドを実行する際の動作手順を説明するためのフローチャートである。

【００３４】実行キューに投入されたスレッドが存在す

7

る場合（図６のステップＢ１のＹ）、オペレーティングシステムは、そのスレッドが属するプロセスのデータ構造１３ａ～１３ｃを調べ（図６のステップＢ２）、メンバp-pgsegpが初期値でない場合には（図６のステップＢ２のＮ）、対応するpgseg データ構造１２ａ～１２ｃのpgseg-pgからprocgroup データ構造１１を求め（図６のステップＢ３）、そのメンバであるpg-cpuを参照して予め設定されたプロセッサＩＤを認識する（図６のステップＢ４）。

【００３５】ここで、オペレーティングシステムは自身のＩＤとそのプロセッサＩＤとが一致しているか否かを判断し（図６のステップＢ５）、一致していればそのスレッドを実行し（図６のステップＢ５のＹ）、不一致であれば、次の実行待ちスレッドについて同様の処理を繰り返す（図６のステップＢ５のＮ）。

【００３６】これにより、図３に示すプロセス９ａ、９ｂ及び９ｃは、同一のプロセッサで実行されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【００３７】次に、図７乃至図９を参照して本発明の第３実施例を説明する。図７及び図８は、同実施例におけるアドレス空間の共有に関するデータ構造８を具体的に示した概念図である。

【００３８】ここでは、図７に示すように、システム中にプロセス９ａ、９ｂ、９ｃ、９ｄ及び９ｅがあり、それらがあるアドレス空間を共有する場合、具体的には、プロセス９ａ、９ｂ、９ｃ及び９ｄがアドレス空間１０ｘを、プロセス９ｄ及び９ｅがアドレス空間１０ｙを、プロセス９ａ及び９ｂがアドレス空間１０ｚをそれぞれ共有する場合を考える。

【００３９】図８に示すように、これらのプロセスに対応するデータ構造は、それぞれ１３ａ～１３ｅである。そして、このプロセス群それぞれに対しprocgroup なるデータ構造１１ｘ～１１ｚを割り当てる。また、このプロセス群に属する各プロセスに対応してpgseg なるデータ構造１２ａ１～１２ｅ１を割り当てる。そして、これらはリンクされた形で保持される。

【００４０】このprocgroup １１ｘ～１１ｚは、pgsegデータ構造を指し示すためのポインタであるpg-pgseglink、これを更新する際に排他制御を行うためのロックであるpg-pgseglock、及びこのprocgroup １１ｘ～１１ｚに属するプロセスが実行されるべきプロセッサのＩＤが格納されるpg-cpuなるメンバにより構成される。一方、pgseg １２ａ１～１２ｅ１には、次のpgseg 構造を指し示すためのポインタであるpgseg-link、同一プロセス内で、別のアドレス空間を共有している場合にそのpgsegデータ構造を指し示すためのポインタであるpgseg-plink 、procgroup１１ｘ～１１ｚを指し示すためのポインタであるpgseg-pg、及びこのpgseg が属するプロセスのデータ構造１３ａ～１３ｅを指し示すためのポインタであるpgseg-procp なるメンバで構成される。また、各プロセスのデータ構造１３ａ～１３ｅには、対応するpgseg データ１２ａ１～１２ｅ１を指し示すためのポインタであるp-pgsegpなるメンバが存在する。このp-pgsegpは、アドレス空間を共有していない場合には初期値が格納されている。また、これらすべてのデータ構造を更新する際に排他制御が必要となるが、そのためのロックを示すものがpglock１４である。

8

【００４１】ここで、図９を参照してプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明する。図９はプロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャートである。

【００４２】初めてアドレス空間の共有が発生した場合には、図５で示した手順により処理を行う。そして、ここでは、既に共有されているアドレス空間に対して新たに共有を要求する場合を説明する。

【００４３】この場合、まず、その新たにアドレス空間の共有を要求したプロセスに対応するpgseg データ構造１２ａ１～１２ｅ１を割り当てる（図９のステップＣ１）。そして、この割り当てたpgseg データ構造と、アドレス空間に対応したprocgroup データ構造１１ｘ～１１ｚとの間をリンクさせ、メンバの初期化を行う（図９のステップＣ２）。

【００４４】ここで、互いに別のアドレス空間を共有しているプロセス同志が新たにアドレス空間を共有する場合には（図９のステップＣ３のＹ）、双方のデータ構造を合わせ、グルーピングする必要がある。この際、アドレス空間の共有を要求した側のprocgroup データ構造の後リンク（pg-forw ）又は前リンク（pg-back ）をたどり、共有する相手側プロセスの属するprocgroup データ構造が出現してこないかを確認する（図９のステップＣ４～Ｃ６）。もし、相手側プロセスの属するprocgroup データ構造が出現した場合には（図９のステップＣ６のＹ）、すでに共有によるデータ構造のグルーピングが行われていることになるので、新たに処理を行う必要はない。一方、相手側プロセスの属するprocgroup データ構造が出現しなかった場合には、グルーピングを行う（図９のステップＣ７）。この場合、要求側プロセスの属するprocgroup データ構造と相手側プロセスの属するprocgroup データ構造とをリンクさせ、要求側プロセスの属するprocgroup データ構造すべてのpg-cpuメンバを相手の属するprocgroup データ構造のpg-cpuメンバに更新する。

【００４５】これにより、図７に示すプロセス９ａ、９ｂ、９ｃ、９ｄ及び９ｅは、同一のプロセッサで実行されることになり、アドレス空間を共有することにより発生するキャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させることができる。

【００４６】次に、図１０を参照して上述したプロセス間のアドレス空間の共有が解消された場合のオペレーティングシステムの動作手順を説明する。図１０はプロセス間のアドレス空間の共有が解消された場合のオペレーティングシステムの動作手順を説明するためのフローチャートである。

【００４７】ここでは、図７の状態から、プロセス９ｄが共有アドレス１０ｘの共有を解消する場合を考える。この場合、まずオペレーティングシステムは、共有の解消するアドレス空間に対応するprocgroup データ構造１１ｘから、pgseg データ構造１２ｄ１を取り除く（図１０のステップＤ１）。この際、そのpgseg データ構造が他のprocgroupデータ構造にも存在する場合には、このpgseg データ構造同士のリンクも適切に更新する。この例では、pgseg データ構造１２ｄ２が存在するため、このpgsegデータ構造１２ｄ２のpgseg-plink を初期化しておく。

【００４８】次に、procgroup データ構造のリンクをたどり、先頭のprocgroup を検索する（図１０のステップＤ２）。この例では１１ｘが該当する。そして、ここにつながれているすべてのpgseg に対して、各々のpgseg-plink をたどり、共有関係のあるprocgroup データ構造を求め、これをprocgroup データ構造のリンクpg-link に加える。この例では、procgroup データ構造１１ｘからpg-pgseglinkをたどり、pgseg １２ａ１がまず検索される。そして、このpgseg １２ａ１は、pgseg１２ａ２に対するリンクをもっているので、pgseg １２ａ２の属するprocgroupデータ構造であるprocgroup データ構造１１ｚをprocgroup データ構造１１ｘのリンクpg-link に加える（図１０のステップＤ４）。オペレーティングシステムは、この動作を繰り返し、pgseg １２ｂ１及びpgseg １２ｃ１についての処理が終了したら、procgroup データ構造１１ｘのpg-link をたどり、次のprocgroupデータ構造について同様の処理を繰り返す。

【００４９】この処理がすべて終了した時点で、プロセス群を分割できる場合には、pg-link でつながれているprocgroup 群とつながれていないprocgroup 群とに分かれるので（図１０のステップＤ６のＹ）、つながれていないprocgroup 群をpg-forw及びpg-back でリンクする（図１０のステップＤ７）。また、つながれているprocgroup 群に対しても、pg-forw 及びpg-back を設定しておくとともに、pg-link を初期化しておく。

【００５０】次に、分割されたグループの一方を、適切なプロセッサで実行させるため、pg-cpuを更新する。なお、pg-link でつながれていないprocgroup 群が存在しなかった場合には、分割ができない場合ということになり、この場合には、pg-form とpg-back を設定してpg-link を初期化する。これにより、システム全体の効率をより向上させることが可能となる。

【００５１】

【発明の効果】以上詳記したように本発明によれば、プロセス間でアドレス空間を共有している場合に、それらのプロセスを同一のプロセッサにスケジューリングして実行することにより、キャッシュメモリや共有メモリの一貫性を保持するためのオーバーヘッドを減少させ、システム性能の向上を図ることが可能となる。

【００５２】また、同一プロセッサで実行すべき必要性がなくなった場合に、実行プロセッサを適切に割り振るため、システム全体の効率をより向上させることが可能となる。

【図面の簡単な説明】

【図１】本発明の第１実施例に係るスケジューリング装置の概略構成を示す図。

【図２】本発明の第２実施例に係るスケジューリング装置の概略構成を示す図。

【図３】第２実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

【図４】第２実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

【図５】プロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャート。

【図６】オペレーティングシステムがスレッドを実行する際の動作手順を説明するためのフローチャート。

【図７】第３実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

【図８】第３実施例におけるアドレス空間の共有に関するデータ構造を具体的に示した概念図。

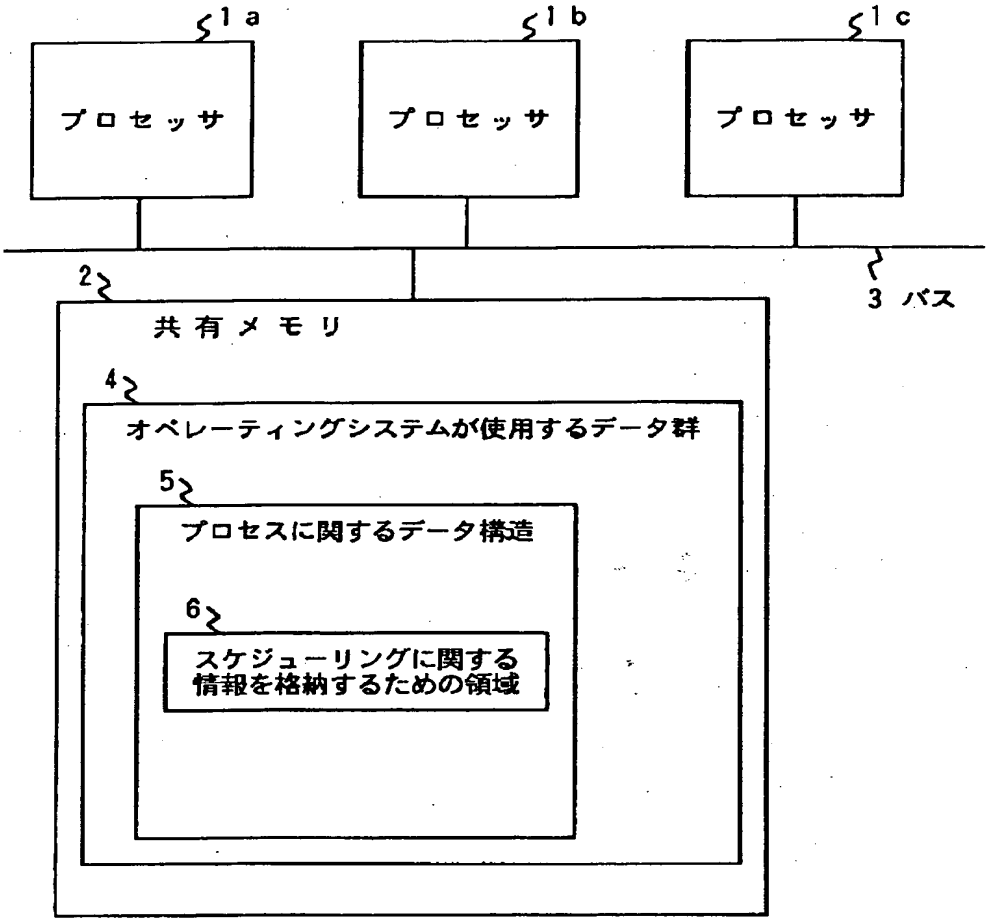
【図９】プロセス間でアドレス空間の共有が起こった場合のオペレーティングシステムの動作手順を説明するためのフローチャート。

【図１０】プロセス間のアドレス空間の共有が解消された場合のオペレーティングシステムの動作手順を説明するためのフローチャート。

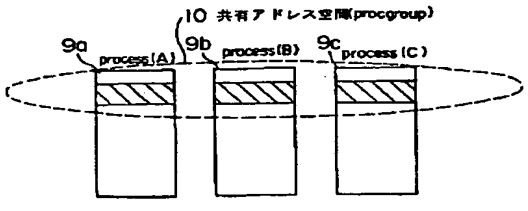
【符号の説明】

１ａ～１ｃ…プロセッサ、２…共有メモリ、３…バス、４…オペレーティングシステムが使用するデータ群、５…プロセスに関するデータ構造、６…スケジュールに関する情報を格納するための領域、７…アドレス空間の共有に関するデータ構造へのポインタ、８…アドレス空間の共有に関するデータ構造。

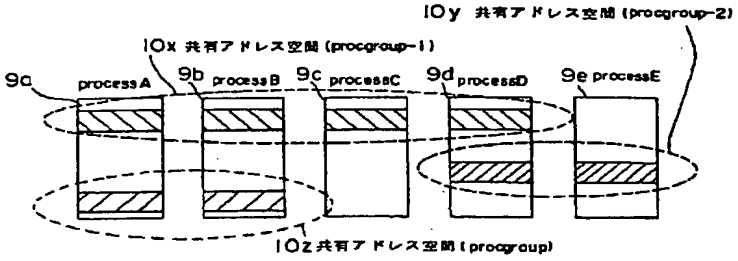
【図1】


1a
1b
1c
プロセッサ
プロセッサ
プロセッサ
2
3 バス
共有メモリ
4
オペレーティングシステムが使用するデータ群
5
プロセスに関するデータ構造
6
スケジューリングに関する情報を格納するための領域

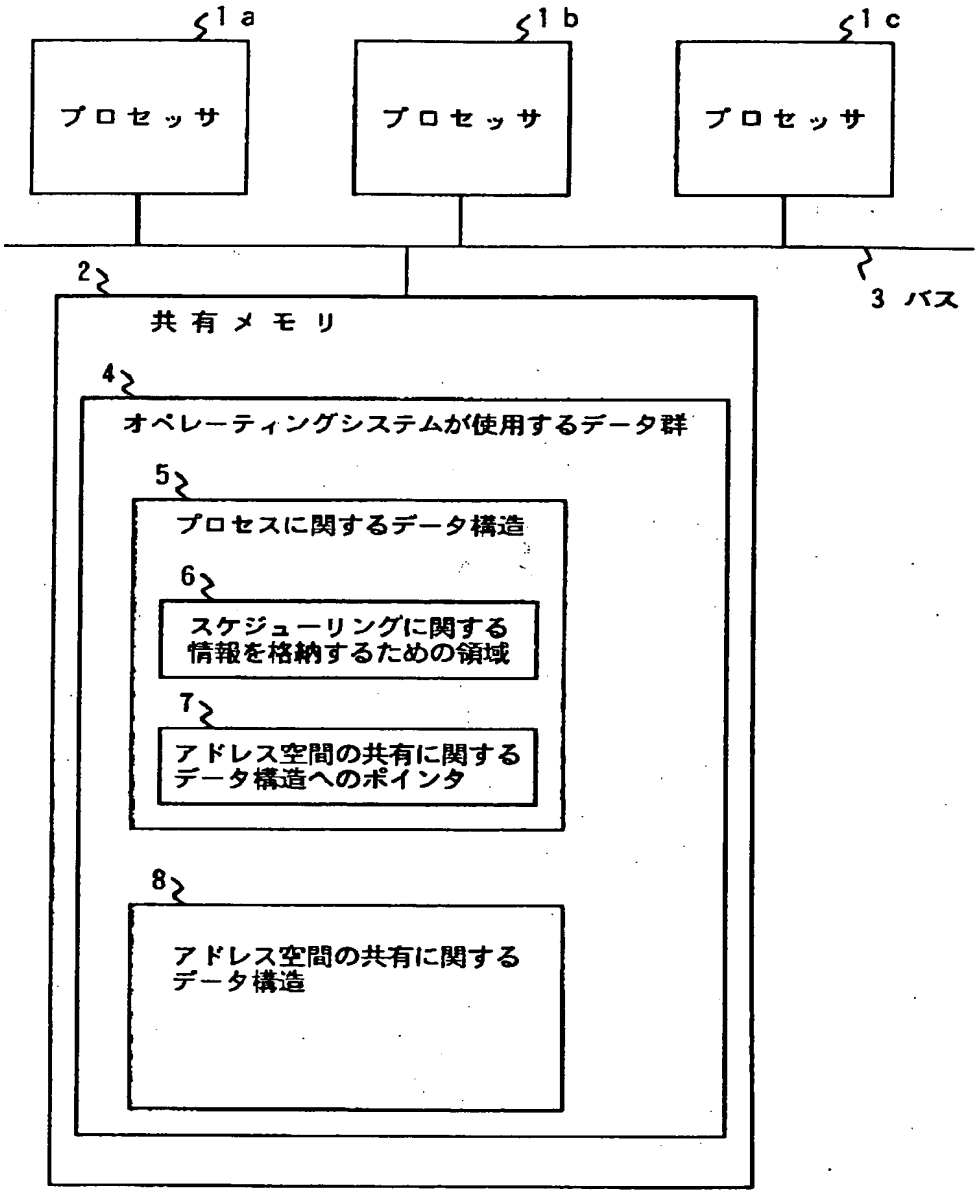
【図3】


10 共有アドレス空間(procgroup)
9a
process(A)
9b
process(B)
9c
process(C)

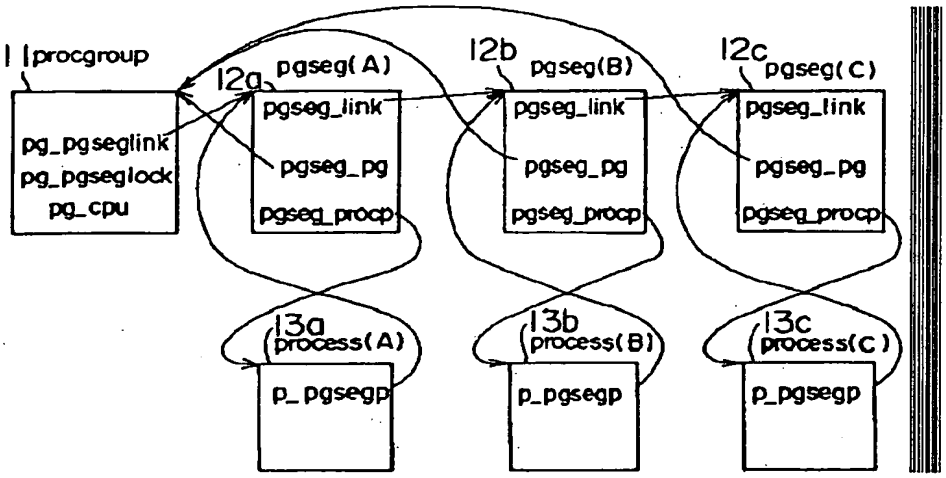
【図7】


10y 共有アドレス空間(procgroup-2)
10x 共有アドレス空間(procgroup-1)
9a
processA
9b
processB
9c
processC
9d
processD
9e
processE
10z 共有アドレス空間(procgroup)

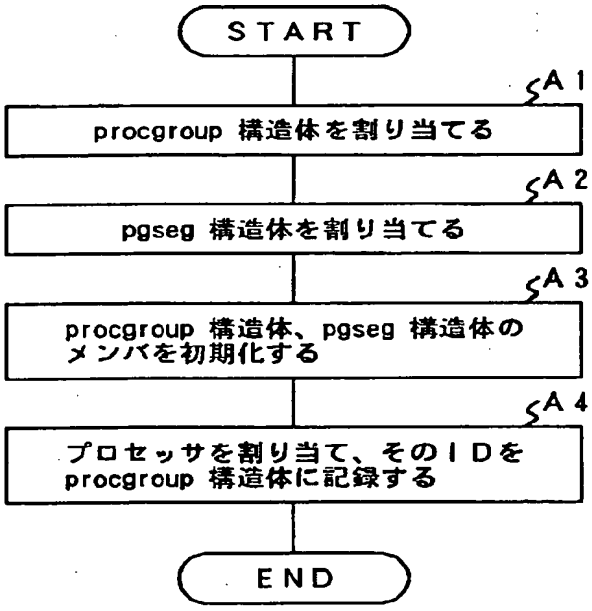
【図2】


1a
1b
1c
プロセッサ
プロセッサ
プロセッサ
2
共有メモリ
3 バス
4
オペレーティングシステムが使用するデータ群
5
プロセスに関するデータ構造
6
スケジューリングに関する情報を格納するための領域
7
アドレス空間の共有に関するデータ構造へのポインタ
8
アドレス空間の共有に関するデータ構造

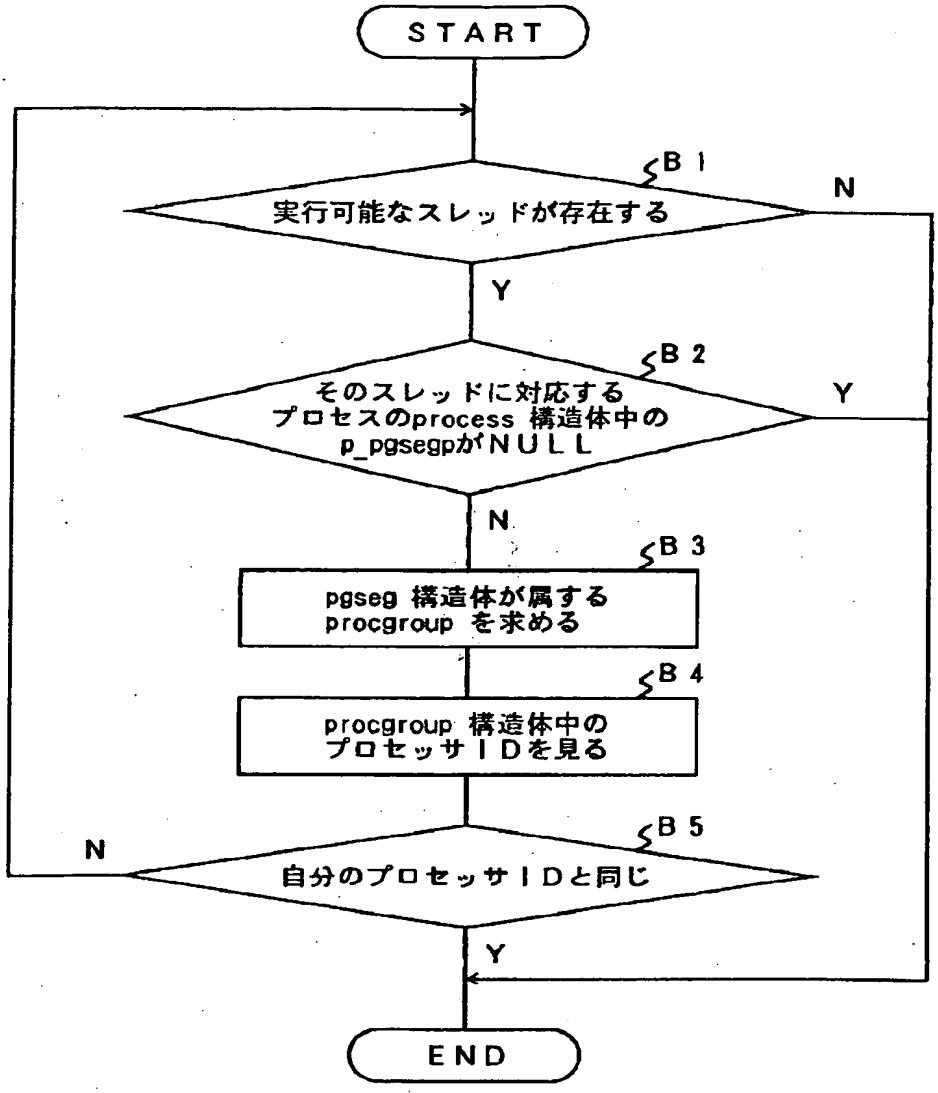
【図4】


11 procgroup
pg_pgseglink
pg_pgseglock
pg_cpu
12a pgseg(A)
pgseg_link
pgseg_pg
pgseg_procp
12b pgseg(B)
pgseg_link
pgseg_pg
pgseg_procp
12c pgseg(C)
pgseg_link
pgseg_pg
pgseg_procp
13a process(A)
p_pgsegp
13b process(B)
p_pgsegp
13c process(C)
p_pgsegp

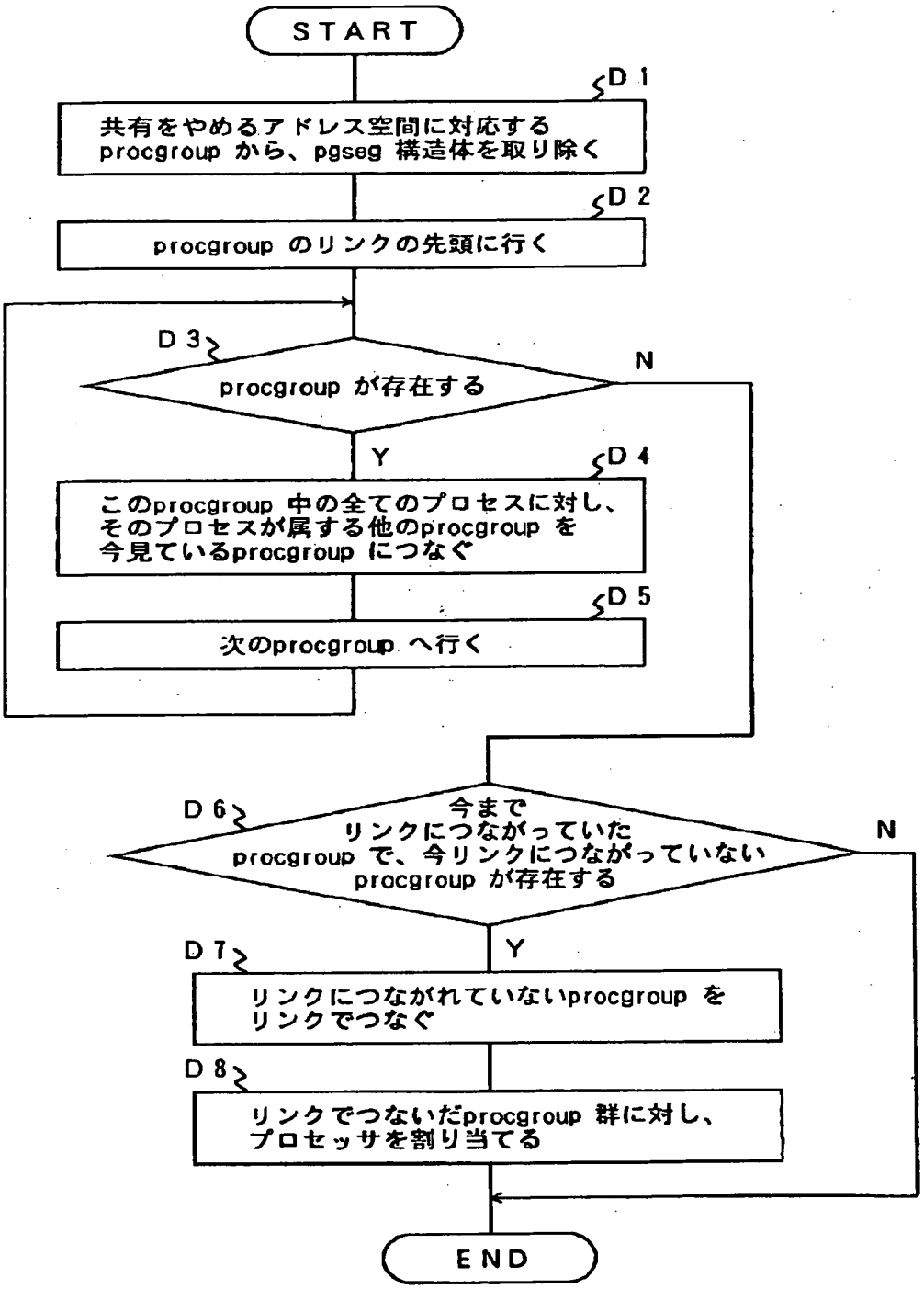
【図5】


START
A1
procgroup 構造体を割り当てる
A2
pgseg 構造体を割り当てる
A3
procgroup 構造体、pgseg 構造体の
メンバを初期化する
A4
プロセッサを割り当て、そのＩＤを
procgroup 構造体に記録する
END

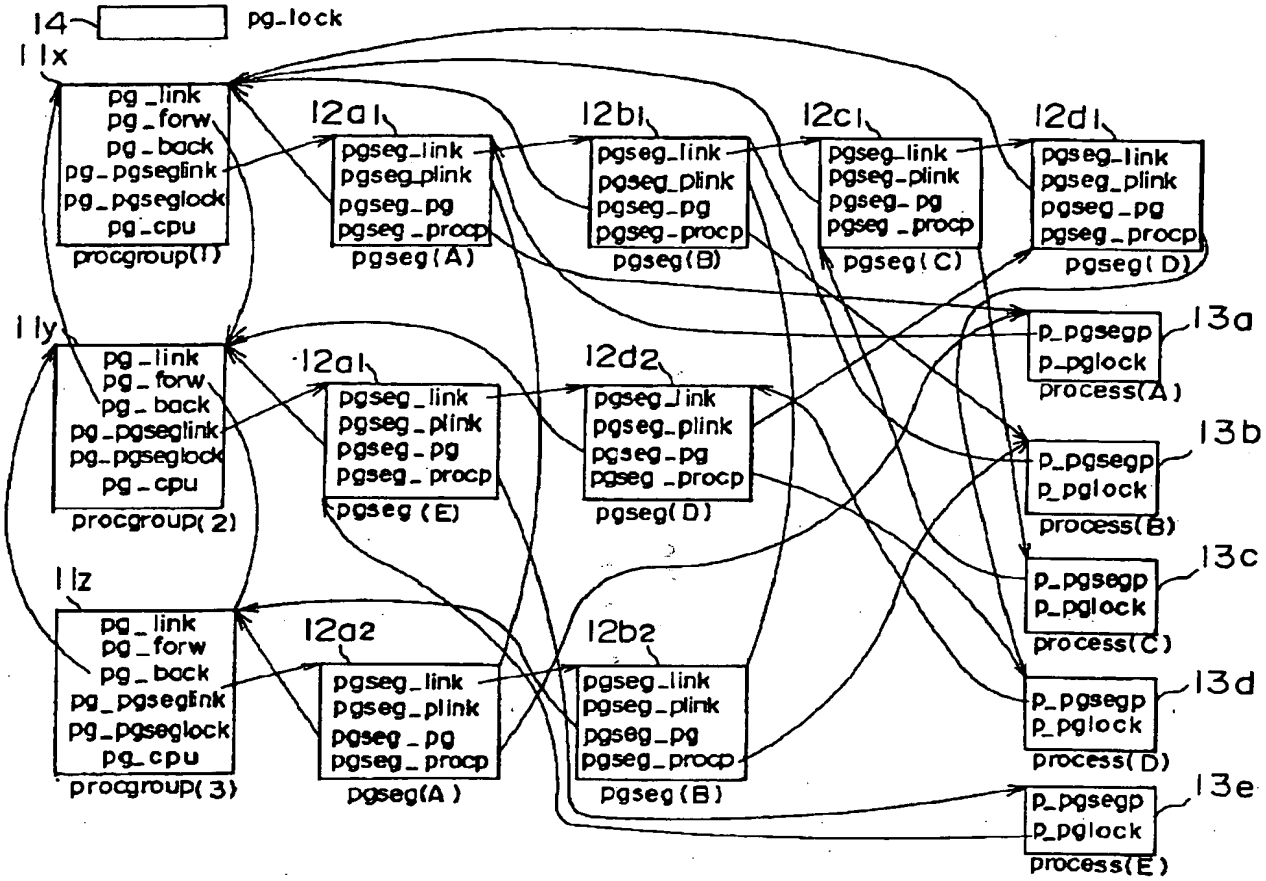
【図6】


START
B 1
実行可能なスレッドが存在する
N
Y
B 2
そのスレッドに対応する
プロセスのprocess 構造体中の
p_pgsegpがＮＵＬＬ
Y
N
B 3
pgseg 構造体が属する
procgroup を求める
B 4
procgroup 構造体中の
プロセッサＩＤを見る
B 5
N
自分のプロセッサＩＤと同じ
Y
END

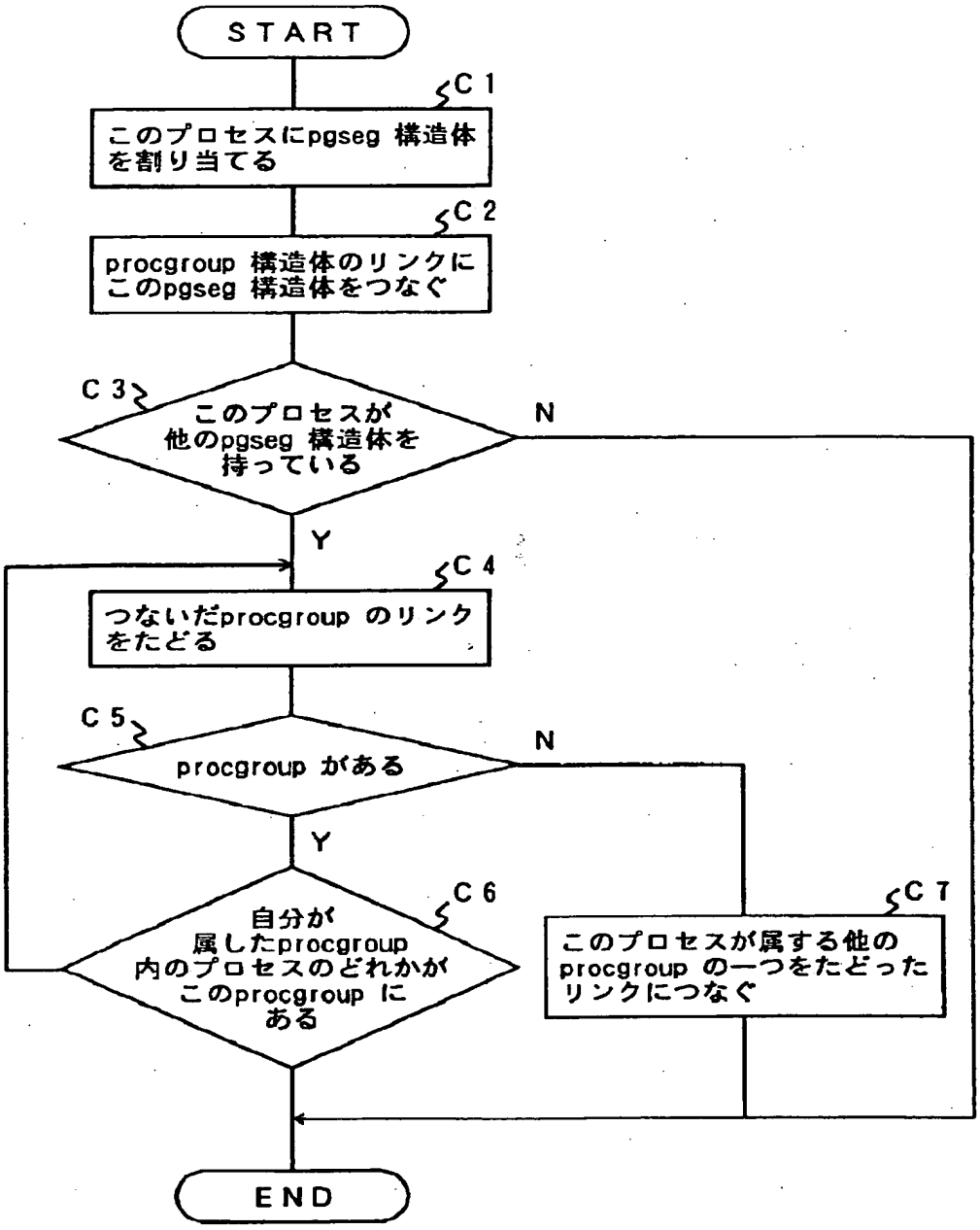
【図8】


START
D 1
共有をやめるアドレス空間に対応する
procgroup から、pgseg 構造体を取り除く
D 2
procgroup のリンクの先頭に行く
D 3
procgroup が存在する
N
Y
D 4
このprocgroup 中の全てのプロセスに対し、
そのプロセスが属する他のprocgroup を
今見ているprocgroup につなぐ
D 5
次のprocgroup へ行く
D 6
今まで
リンクにつながっていた
procgroup で、今リンクにつながっていない
procgroup が存在する
N
Y
D 7
リンクにつながれていないprocgroup を
リンクでつなぐ
D 8
リンクでつないだprocgroup 群に対し、
プロセッサを割り当てる
END

【図9】


14
pg_lock
11x
pg_link
pg_forw
pg_back
pg_pgseglink
pg_pgseglock
pg_cpu
procgroup(1)
12a1
pgseg_link
pgseg_plink
pgseg_pg
pgseg_procp
pgseg(A)
12b1
pgseg(B)
12c1
pgseg(C)
12d1
pgseg(D)
11y
procgroup(2)
12a1
pgseg(E)
12d2
11z
procgroup(3)
12a2
12b2
13a
p_pgsegp
p_pglock
process(A)
13b
process(B)
13c
process(C)
13d
process(D)
13e
process(E)

【図１０】


START
C 1
このプロセスにpgseg 構造体を割り当てる
C 2
procgroup 構造体のリンクにこのpgseg 構造体をつなぐ
C 3
このプロセスが他のpgseg 構造体を持っている
N
Y
C 4
つないだprocgroup のリンクをたどる
C 5
procgroup がある
N
Y
C 6
自分が属したprocgroup 内のプロセスのどれかがこのprocgroup にある
C 7
このプロセスが属する他のprocgroup の一つをたどったリンクにつなぐ
END